Curso 09/10

Curso de Enseñanzas Propias
Introducción a **GNU/Linux**

UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA

Sesión 3
Comandos avanzados

Mª José Santofimia
Escuela Superior de Informática
Universidad de Castilla-La Mancha

molinux

Contenido

- Utilidades de Terminal
- Tareas

- Entrada/Salida Estándar
- Pipes y Redirección

Utilidades de Terminal

- grep

- find
- Variables de entorno

grep - *Global Regular Expression Print*

- Búsqueda global en función de expresiones regulares

http://osr507doc.sco.com/en/OSTut/graphics/grep.gif

- Uso

```
grep [opciones] expresión regular [archivo]
```

grep. Opciones

`-c,--count:` Suprime la salida normal; en su lugar muestra el número de líneas que concuerdan con el patrón para cada fichero de entrada.

`-e patrón,--regexp=PATRÓN:` Emplea patrón como el patrón; útil para proteger patrones que comiencen con -.

`-f fichero,--file=FICHERO:` Obtiene el patrón de fichero.

`-i,--ignore-case:` No hace caso de si las letras son mayúsculas o minúsculas ni en el patrón ni en los ficheros de entrada.

grep. Opciones

`-v,--revert-match:` Invierte el sentido de la concordancia, para seleccionar las líneas donde no las hay.

`-w,--word-regexp:` Selecciona solamente aquellas líneas que contienen concordancias que forman palabras completas. La comprobación consiste en que la cadena de carácteres concordante debe estar al principio de la línea o precedida por un carácter que no forme parte de una palabra. De forma similar, debe estar o al final de la línea o ser seguida por un carácter no constituyente de palabra. Los caracteres que se consideran como parte de palabras son letras, dígitos y el subrayado.

`-s,--silent:` Suprime los mensajes de error sobre ficheros que no existen o no se pueden leer.

Meta-carácteres y Expresiones Regulares

- **Meta-carácteres**
  - `*` cero o más letras
  - `?` un solo carácter
  - `[ ]` lista de carácteres posibles
  - `-` rango de valores
  - `\` ignorar el meta-carácter siguiente

- **Expresiones Regulares**
  - `^` al principio de la línea
  - `$` al final de la línea
  - `[ ]` lista de carácteres posibles
  - `-` rango de valores
  - `\` ignorar el meta-carácter siguiente
  - `[^expr]` cualquier expresión excepto *expr*
  - `r*` cero o más ocurrencias de r

grep. Ejemplos de uso

- Búsqueda de la ocurrencia de una palabra en un fichero.

```
$ grep palabra fichero
```

- Sin distinción entre mayúsculas y minúsculas.

```
$ grep -i palabra fichero
```

- Búsqueda recursiva sobre todos los ficheros de un directorio.

```
$ grep -r palabra ruta
```

- Cuenta el número de ocurrencias de una palabra en un determinado fichero.

```
$ grep -c palabra fichero
```

grep. Ejemplos de uso

- Busca las líneas que no contengan una determinada palabra.

```
$ grep -v palabra fichero
```

- Lista el nombre de aquellos archivos, de un determinado tipo, que contengan una cadena.

```
$ grep -l palabra *.ext
```

```
$ grep -l "main()" *.c
```

- Para que la búsqueda resulte más agradable a la vista, se pueden utilizar colores para resaltar.

```
$ grep --color palabra fichero
```

grep. Ejemplos de uso

- Búsqueda de palabras completas, evitando la aparición en subcadenas.

```
$ grep -w palabra fichero
```

- Mostrar las líneas que comienzan con una secuencia de caracteres determinado.

```
$ grep '^seq' fichero
```

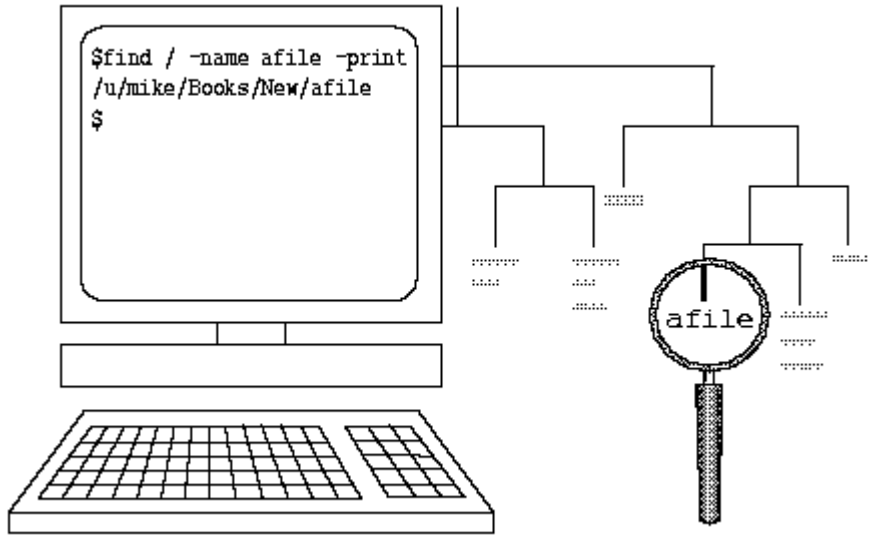
- Búsqueda de líneas que contengan alguna de las siguientes palabras.

```
$ grep -e '(plb1|plb2)' fichero
```

find

- Búsqueda de un archivo en una jerarquía de directorios
- Uso

```
find [ruta] [expresión]
```



http://osr507doc.sco.com/en/OSTut/graphics/find.gif

find. Principales Opciones

-name <*expresion*>: especifica el patrón del nombre de archivos a buscar.

-type: especifica que tipo de fichero se quiere buscar. Los tipos de fichero que se le pueden poner para buscar son los siguientes:

- **d -** directorios
- **f -** ficheros
- **l -** enlaces simbólicos
- **s –** sockets

-size +/-<*n*>: busca ficheros de un determinado tamaño (n en KiB).

-exec <*comando*> {};: Comando a ejecutar con los ficheros resultados de la búsqueda. Los {} se sustituyen por los ficheros encontrados

find. Ejemplos de uso

- Listado de todos los ficheros con extensión .c que cuelgan de un determinado directorio.

```
$ find /directorio -name '*.c'
```

- Listado de los ficheros que cuelguen del home de usuario “alumno” y contengan la subcadena “practica”.

```
$ find /home/alumno -name '*practica*'
```

- Búsqueda de los ficheros con extensión .c y compilación.

```
$ find /home/alumno/practica1 -name '*.c' -exec gcc {} \;
```

- Búsqueda de aquellos ficheros de más de 500MiB.

```
$ find /home/alumno -size +500000k
```

Variables de entorno

- Las variables de entorno se utilizan para proporcionar argumentos de configuración a los programas.
- El CLASSPATH de java es un ejemplo de variable de entorno.

Variables de entorno

- El comando **`env`** muestra la lista de variables de entorno definidas en nuestro sistema.
- El comando **`set`**, también las muestras, aunque se utiliza para definir las variables.
- El comando **`unset`** se utiliza para eliminar la definición de una variable de entorno.
- El comando **`export`**, si no está definida la variable, la define; si lo está, sustituye su valor.
- El comando **`echo`** muestra el contenido de una variable.

Variables de entorno. Ejemplos

- Muestra el contenido de la variable PATH.

```
$ echo $PATH
```

- Crea un archivo ls y escribe una línea de texto.

```
$ echo "echo \"es ls no hace nada\"" > texto.txt
```

- Dale permisos de ejecución.

```
$ chmod +x ls
```

- Modifica la variable PATH para que busque primero en el directorio de trabajo.

```
$ export PATH=/home/alumno:$PATH
```

- Ejecuta el comando ls ¿qué ocurre?

Planificando Tareas

- Las tareas de administración suelen consistir en tareas rutinarias.
- Sistemas para la construcción de "*agendas de tareas*":
  - `cron`
  - `at`

http://www.flickr.com/photos/81828336@N00/2914061212/

cron

- Establece la programación de una lista de tareas a ejecutar.
- Las tareas están configuradas como entradas del fichero /etc/crontab:
  - Minutos y hora en que se lanzará la tarea.
  - Día del mes.
  - Mes.
  - Día de la semana.
  - Tarea a ejecutar (puede ser un directorio donde se encuentren las tareas a ejecutar).

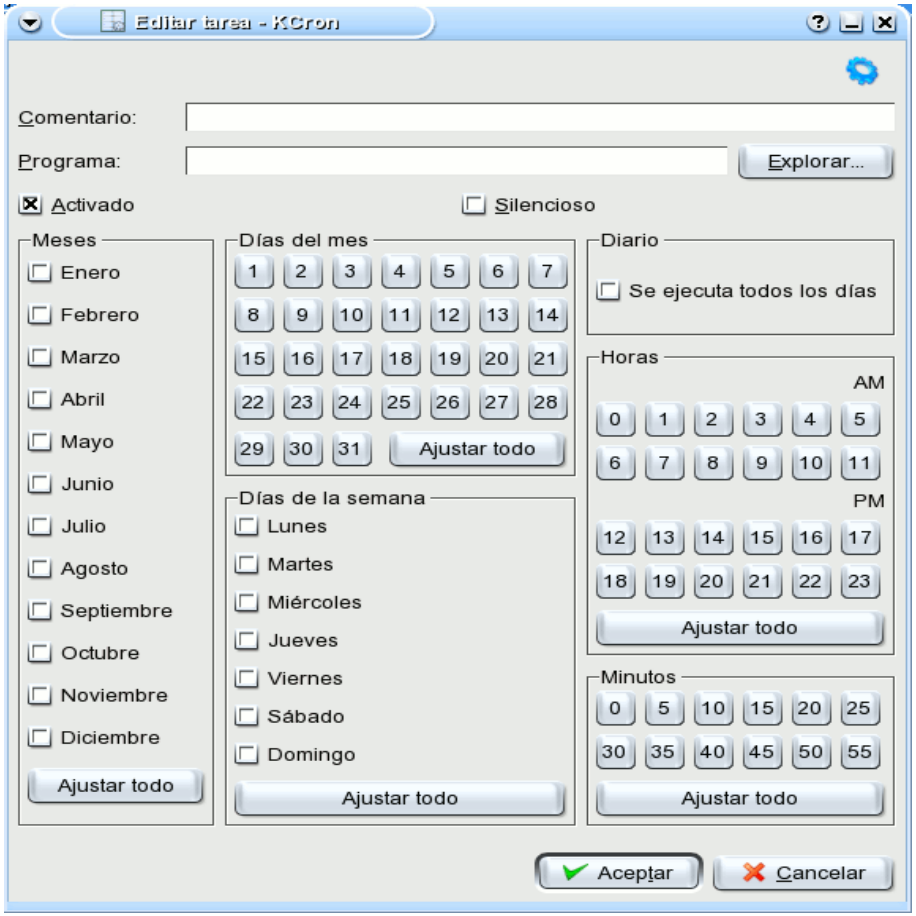
cron. Ejemplos

- Crea una copia de seguridad tar.gz del contenido de tu directorio de trabajo cada domingo de la semana.
- Programa un despertador de lunes a viernes a las 7.30 de la mañana.
- Programa una tarea que avise cuando queden 30 minutos de clase (13.30h).

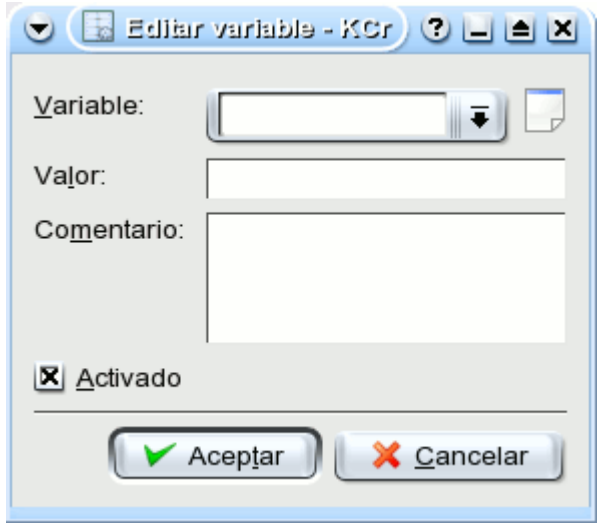
```
00 5   7 *  *     tar cfz /tmp/copiaDeSeguridad.tar.gz /home/alumno/backup
30 7   *  *  1-5 DISPLAY=":0" totem /home/alumno/musica/Amelie/*.mp3
30 13 *  *   *    echo "quedan 30 minutos de clase"
```

kcron

- Interfaz gráfica para cron.



- También para las variables de entorno.



at

- Permite programar una tarea, una vez, para una fecha y hora concreta.
- La fecha admite “*today*” y “*tomorrow*”.
- Por defecto sólo ejecuta aplicaciones de consola. DISPLAY=:0 permitirá lanzar aplicaciones gráficas.
- El resultado de la ejecución de la tarea se puede redirigir a un archivo para conocer el resultado.

at. Ejemplos de uso

- Reproducir una lista de canciones dentro de 10 minutos.

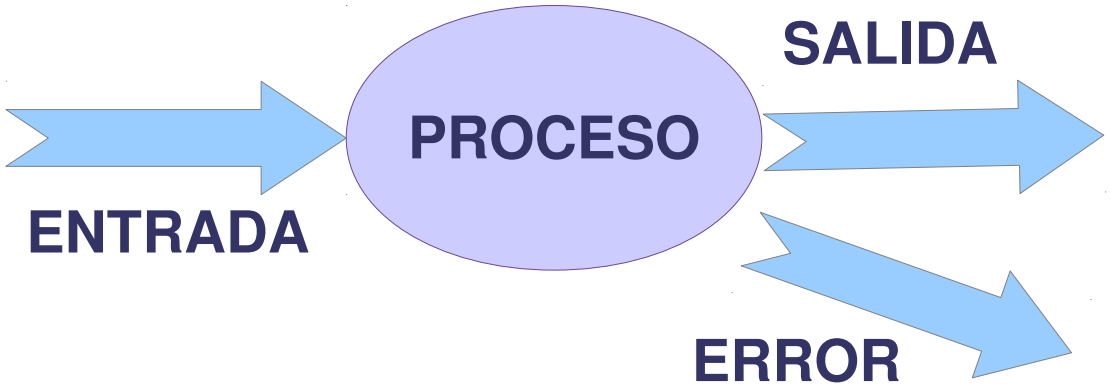
```
$ at now +10 minutes
warning: commands will be executed using /bin/sh
at> DISPLAY=:0 totem /home/alumno/musica/Disc1/*.mp3
at> (Ctrl+D) <EOT>
```

- Apaga el ordenador a las 14h de hoy.

```
$ at 14:00
warning: commands will be executed using /bin/sh
at> halt
at> (Ctrl+D) <EOT>
```

E/S Estándar

- En la ejecución de un proceso (un comando por ejemplo) se diferencia tres flujos:



- El intérprete de comandos configura el sistema de forma que la entrada estándar es el teclado y la salida la pantalla (ej. comandos cat y sort).

E/S Estándar

- La entrada estándar (o *stdin*), por defecto se refiere al teclado. STDIN se identifica con el valor 0.
- La salida estándar (o stdout), por defecto se refiere a la pantalla. STDOUT se identifica con el valor 1.
- El error estándar (o stderror), que de manera predeterminada se refiere a la pantalla, se identifica con el valor 2.

Redireccionamiento

- El intérprete de comandos nos permite redireccionar la salida estándar a un fichero.
- Redireccionar la impresión del mensaje “Hola, Mundo”, impreso dentro de 3 minutos al fichero /tmp/hola.

```
$ at now +3 minutes
Warning: commands will be executed using /bin/sh
at> echo “Hola, Mundo” > /tmp/hola
at> (^D) <EOT>
$ cat /tmp/hola
```

Tipos de Redireccionamiento

- Redireccionamiento destructivo (si el fichero ya existe, lo sobre-escribe) utilizando >
- Redireccionamiento no destructivo (se añade al final del archivo) utilizando >>
- Redireccionamiento de la salida de error, utilizando 2>

Redireccionar la salida de error

- Ejecuta el comando grep de manera incorrecta (por ejemplo con una opción que esté definida) y redirecciona la salida de error a un fichero.

```
$ grep -k 2> error.txt
$ cat error.txt
grep: opción inválida -- k
Modo de empleo: grep [OPCIÓN]... PATRÓN [FICHERO]...
Pruebe `grep --help' para más información.
```

Redireccionamiento. Ejemplos

- Crea la lista de la compra con los siguientes elementos:

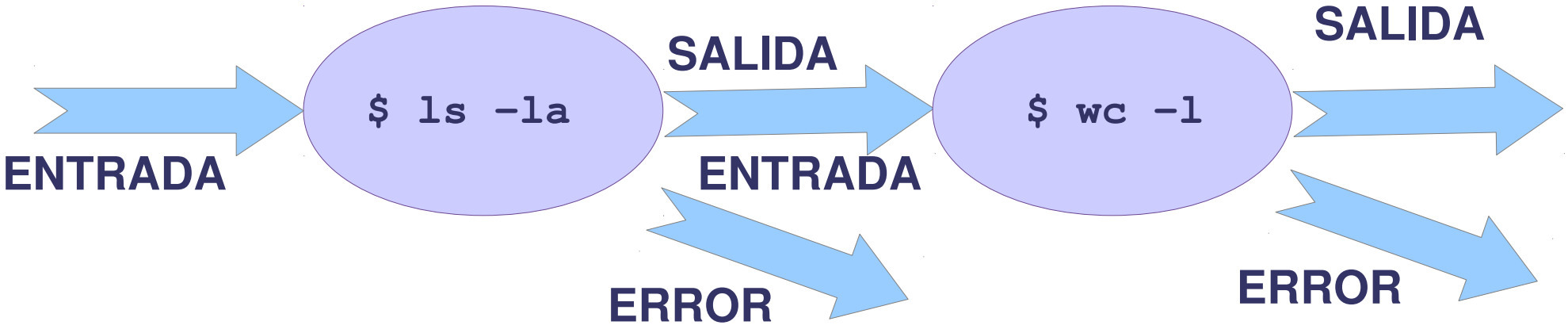
```
$ cat > lista_compra
platanos
manzanas
leche
pan <Ctrl-D>
```

- Ordena la lista y salva la nueva como lista_ordenada:

```
$ sort < lista_compra > lista_ordenada
```

Tuberías (pipeline)

- Son un mecanismo de comunicación que permite enlazar la salida de un comando con la entrada de otro.



Tuberías. Ejemplos de uso

- Contar el número de archivos de un directorio.

```
$ ls -la | wc -l
```

- Ordenar los archivos alfabéticamente.

```
$ ls -la | sort
```

- Mostrar los archivos cuyo nombre comienza por “lis”.

```
$ ls -la | grep 'lis'
```

Caso práctico

- Tenemos una colección de archivos de música (en formato mp3), organizados y nombrados en una estructura como la siguiente:

GRUPO - Album

01 – Nombre del tema.mp3
02 – Nombre del tema.mp3
.
.
.
12 – Nombre del tema.mp3